REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From - To)*
10-05-2006	Journal Paper PREPRINT	2006

4. TITLE AND SUBTITLE
Autonomous Distant Visual Surveillance of Satellites
(PREPRINT)

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
John E. McInroy, Lawrence M. Robertson,* R. Scott Erwin*

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Dept of Electrical and Computer Engineering
University of Wyoming
Laramie, WY 82071

*Air Force Research Laboratory
Space Vehicles
3550 Aberdeen Ave SE
Kirtland AFB, NM 87117-5776

8. PERFORMING ORGANIZATION REPORT NUMBER
AFRL-VS-PS-JA-2006-1016

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/VSSV

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited. (Clearance #VS06-0188)

13. SUPPLEMENTARY NOTES
Submitted for publication in IEEE Transactions on Aerospace and Electronic Systems

Government Purpose Rights

14. ABSTRACT
This paper develops three new, interconnected techniques useful for the autonomous distant visual inspection of satellites. First, silhouetting of man made, erratically illuminated satellites is performed. Illumination cases include full sun from an arbitrary (often awkward) viewing angle and un-illuminated (back-lit by the star field). New Statistical Straight Line Snakes (SSLS) prove efficient in finding the silhouette, even in the un-illuminated case. The silhouette is composed of straight line segments, which are easy to calculate, fit the straight lines inherent in man made objects, and lend themselves to further processing (pose estimation, template matching, etc.). Once the silhouette has been used to find correspondence points, a second method for detecting a moving, nearby chaser vehicle is derived. The hard case is treated in which the chaser and satellite are so nearby that their images are blurred together. The algorithm finds the dimension of motion generated by the sequence of images. If the dimension is higher than that explained by a single rigid body, then this indicates a possible chaser. No knowledge of the satellite or chaser's shape or motion is required. Independent relative motion between the satellite and chaser is required–if the chaser is immobile with respect to the satellite, then a third technique must be used. This third method incorporates the satellite's solid model to estimate its pose from a noisy, diffraction limited image. The pose is then combined with the solid and optical model to create synthetic expected images. Inspection is performed by comparing these with the actual images. The new pose algorithm first estimates depth by a least upper bound technique. A fast method is derived of optimally estimating the rotation matrix by a sequence of analytical solutions (rather than a nonlinear numerical optimization!). Simulations illustrate the use of all three techniques on images obtained when viewing low Earth orbit satellites from the ground.

15. SUBJECT TERMS
Affine Cameras, Pose Estimation, Visual Servoing, Template Matching, Object Recognition, Motion Subspace

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**	Unlimited	11	Lawrence Robertson
Unclassified	Unclassified	Unclassified			**19b. TELEPHONE NUMBER** *(include area code)* 505-846-7687

Autonomous Distant Visual Surveillance of Satellites

John E. McInroy, *Senior Member, IEEE,* Lawrence M. Robertson and R. Scott Erwin

***Abstract*— This paper develops three new, interconnected techniques useful for the autonomous distant visual inspection of satellites. First, silhouetting of man made, erratically illuminated satellites is performed. Illumination cases include full sun from an arbitrary (often awkward) viewing angle and un-illuminated (back-lit by the star field). New Statistical Straight Line Snakes (SSLS) prove efficient in finding the silhouette, even in the un-illuminated case. The silhouette is composed of straight line segments, which are easy to calculate, fit the straight lines inherent in man made objects, and lend themselves to further processing (pose estimation, template matching, etc.) Once the silhouette has been used to find correspondence points, a second method for detecting a moving, nearby chaser vehicle is derived. The hard case is treated in which the chaser and satellite are so nearby that their images are blurred together. The algorithm finds the dimension of motion generated by the sequence of images. If the dimension is higher than that explained by a single rigid body, then this indicates a possible chaser. No knowledge of the satellite or chaser's shape or motion is required. Independent relative motion between the satellite and chaser is required– if the chaser is immobile with respect to the satellite, then a third technique must be used. This third method incorporates the satellite's solid model to estimate its pose from a noisy, diffraction limited image. The pose is then combined with the solid and optical model to create synthetic expected images. Inspection is performed by comparing these with the actual images. The new pose algorithm first estimates depth by a least upper bound technique. A fast method is derived of optimally estimating the rotation matrix by a sequence of analytical solutions (rather than a nonlinear numerical optimization!). Simulations illustrate the use of all three techniques on images obtained when viewing low Earth orbit satellites from the ground.**

***Index Terms*— affine cameras, pose estimation, visual servoing, template matching, object recognition, motion subspace**

I. Introduction

Visual surveillance of satellites presents new challenges and opportunities for computer vision. The distance between an Earth based telescope and a satellite is great, which implies that the images are often blurry due to the effects of the atmosphere, diffraction, etc. Moreover, illumination is erratic and harsh, varying from a very sharp contrast, to partial illumination, to un-illuminated but back-lit by stars. Finally, the objectives are somewhat different. For instance, it may be of interest just to detect that any body is in the image. Alternatively, it may be desired to know if a single blurry image contains multiple bodies.

This work was supported by the National Research Council and Air Force Office of Scientific Research.

John E. McInroy is with the Department of Electrical and Computer Engineering, University of Wyoming E-mail:mcinroy@uwyo.edu, Phone: (307) 766-6137 , fax: (307)766-2248.

L. Robertson and S. Erwin are with the Air Force Research Laboratory, Space Vehicles Directorate, AFRL/VSSV, Bldg. 472 Kirtland AFB, NM, 87117-5776, E-mail:lawrence.robertson@kirtland.af.mil, richard.erwin@kirtland.af.mil.

This paper develops three new, complementary tools for the distant visual surveillance of satellites. First, silhouetting of man made, erratically illuminated space objects will be performed. Second, methods for detecting multiple independently moving, visually overlaid bodies will be found. Third, methods for inspecting known satellites will be derived.

Finding the silhouette of spaceborne objects is useful for both detecting the presence of an object and for finding correspondences between images and models. When viewing from Earth, sun illumination plays a significant role, as it can be very harsh, partial, or even completely missing. The un-illuminated case is particularly unique, as much can still be done due to back lighting by stars. Traditional gradient based edge detectors ([9], [10]) fail utterly in the unilluminated case. More recent contour growing ("snake") techniques ([17], [14], [1], [3]) are more promising. This paper develops a new Statistical Straight Line Snake (SSLS) algorithm which is simple and naturally fits man made objects with their preponderance of straight edges. It directly outputs lines and vertices ready for use in correspondence, pose, and template matching algorithms.

Nearby moving objects are always a concern for satellites. Unfortunately, visual evaluation from Earth can be difficult due to blurry images caused by diffraction, atmospheric, and scaling effects. Satellites are in constant motion: is the motion seen in an image caused by a single satellite, or are multiple bodies present? This paper modifies a robotic motion segmentation algorithm [18] to answer this question. No model of the motion, satellite, or other objects is required.

On the other hand, relative motion between the multiple bodies is required. To inspect known satellites, and especially to detect bodies immobile with respect to the satellite, a new affine camera pose estimation algorithm is derived. Once the pose is found, it is combined with optical models and solid satellite models to predict the image. Large differences indicate an anomaly.

Despite the availability of many pose estimation algorithms (see [11] [4] [6] [15], [8], and [2] for starters), there remains a need for a rapid, optimal, dependable algorithm suitable for a small number of point correspondences (but more than $n = 4$). For instance, [16] recently added high bandwidth inertial sensing to complement the available pose estimation because it alone had too low of a bandwidth. For the case of affine imaging, this paper develops a new method that is closed form, yet provides optimal estimates even when $n > 4$. It does so by solving the standard 3d-3d optimal orientation problem 2^n times, where n is the number of data points. Since the optimal orientation problem can be very quickly solved as a 4 x 4 matrix eigenvalue problem, the overall computational burden is light. A least upper bound estimate of the scaling (depth) is proposed and combined with the orientation algorithm to find

the complete pose.

This paper explains these techniques in roughly the same order they would typically be performed. First, a method of finding the silhouette will be described. It outputs point correspondences that are needed by both of the other algorithms. This data can be directly used for determining if the sequential images contain multiple bodies. Further inspections can be performed by incorporating the satellite's known shape.

II. Silhouette of Man Made, Erratically Illuminated Objects in Space

This paper seeks to develop a technique for silhouetting which builds on existing methods, but modifies them to be especially robust when finding point or line correspondences on man made objects under the erratic illumination found in space. This may include partial illumination, zero illumination against a star filled background, harsh illumination, and bright reflections. The method is inspired by contour (or snake) growing ([17], [14], [1]), but exploits the inherent straight line structure dominant in man made objects such as satellites, as well as the specular nature of the star filled background. Rather than fitting an arbitrary contour around an object, it forms a contour from line segments. The length and number of the segments is iteratively adapted to fit the measurements.

A brief outline of the steps are as follows:

1) Place an initial, small polygon inside the object's image. Store it as a list of vertex points proceeding in a counter-clockwise direction around the polygon.
2) Eliminate a vertex if the line between adjacent vertices is shorter than the desired resolution.
3) Add vertices to keep each edge a length l_d or shorter.
4) Create a set of points along a straight line edge between vertices. If a point is inside the object, then perturb that point outward along the edge normals. Repeat this for all edges along the polygon.
5) Fit lines to these perturbed points along each edge.
6) Create new vertices by intersecting non-collinear adjacent lines.
7) Check for convergence, returning to step 2 until convergence takes place.

The steps will now be covered in more detail.

A. Step 1: Place an initial, small polygon inside the object's image. Store it as a list of vertex points proceeding in a counter-clockwise direction around the polygon.

An initial point inside the object's image can be found by creating a mask which encodes the object's rough shape, then moving it over the image until the best fit is found. The centroid of the mask giving the best match is the initial point. A small circular polygon (like a hexagon or octagon) is then placed inside the object, centered at the initial point. The number of sides in this initial polygon is not especially important, as vertices will be added and removed during the iterations to enforce edge length and non-collinearity specifications.

B. Step 2: Eliminate a vertex if the line between adjacent vertices is short.

Let $\vec{d}_j$ and $\vec{d}_{j+1}$ (both in $\mathbb{R}^2$) denote two adjacent vertices. The length of edge j can be found from $l_j = ||\vec{d}_{j+1} - \vec{d}_j||$. If this length is below the desired resolution (l_{min}), then vertex $j+1$ is dropped from the list of vertices. This process intentionally limits the resolution of the contour fit, which is useful, for instance, to prevent noise from creating false edges.

C. Step 3: Add vertices to keep each edge a length l_d or shorter.

Let l_d be a desired target length for each edge. If $l_j > l_d$, then a new vertex can be introduced between $\vec{d}_j$ and $\vec{d}_{j+1}$ at the midpoint. This is performed by inserting the additional vertex, $(\vec{d}_j + \vec{d}_{j+1})/2$. The length of this segment (from $\vec{d}_j$ to the new $\vec{d}_{j+1}$) can then be evaluated, and if larger than l_d, another vertex can be added at the midpoint of $\vec{d}_j$ and the newly created $\vec{d}_{j+1}$. This process is repeated until the length between $\vec{d}_j$ and the next vertex is less than l_d, then j is incremented. The distance between the (possibly newly created) vertex $j+1$ and the (possibly newly created) vertex $j+2$ is compared to l_d, and so forth. The process repeats until all edges have a length less than l_d.

In the next step, new edges will be found by perturbing points along each edge until they reach the boundary, then fitting a line to each edge from these perturbed points. By keeping the edges from growing too large, excessive averaging is avoided, thus allowing the contour to conform to the boundary.

D. Step 4: Create a set of points along a straight line edge between vertices. If a point is inside the object, then perturb that point outward along the edge normals.

The unit vector pointing between adjacent vertices is

$$\vec{v} = \frac{\vec{d}_{j+1} - \vec{d}_j}{||\vec{d}_{j+1} - \vec{d}_j||}$$

Points, $\vec{x}$, along this line segment can be created by $\vec{x} = \vec{d}_j + \alpha\vec{v}$ where α is between 0 and the length of the line segment, l_j. Let $\vec{u}$ denote the outward pointing normal. Since the vertices are in a counter-clockwise direction, the cross product $\vec{u} \times \vec{v}$ is positive. This implies that

$$\vec{u} = \begin{bmatrix} v_2 \\ -v_1 \end{bmatrix}$$

The points can then be perturbed along the edge normal by

$$\vec{x} + (p(I(\vec{x})|O) - p(I(\vec{x})|B))g\vec{u} \rightarrow \vec{x} \qquad (1)$$

where $p(I(\vec{x})|O)$ denotes the probability that intensity $I(\cdot)$ occurs at location $\vec{x}$, given that the point lies on the object, $p(I(\vec{x})|B)$ denotes the probability that intensity $I(\cdot)$ occurs at location $\vec{x}$, given that the point lies on the background, and g is a positive scalar gain.

Since the polygon is initially placed somewhere deep inside the object, initially it is desirable to have a large gain, g, that will perturb the data points several pixels per iteration.

Once the edges approach the boundary, then the gain should be reduced so that a fine convergence to the edges will result. There are many possible functions which perform this action. One method is to make g a function of the iteration number, decreasing it linearly as the iterations proceed. Another method is to make g a function of $e = p(I(\vec{x})|O) - p(I(\vec{x})|B)$; e is between -1 and 1. Since the polygon is initially inside the object, e starts at $e = 1$. Once the polygon is close to the boundary, $e \to -1$. To begin with a large gain, g_{max}, which exponentially decreases with e and ends (when $e = -1$) with the small gain, g_{min}, let

$$g(e) = a + e^{be} \tag{2}$$

where

$$b = \text{arcsinh}\frac{g_{max} - g_{min}}{2},\ a = g_{max} - e^b \tag{3}$$

The star-filled backdrop in space images creates unique problems as well as opportunities. When the satellite is illuminated, the main problem is the stars act as a source of noise, making it hard to tell (at distinct points) where the spacecraft ends and the stars begin. The straight line averaging explained in the next section has proven robust to this noise.

On the other hand, the satellite is back lit by stars, thus it forms a (dark) image when it is not illuminated by the sun. This provides an opportunity to image the silhouette even without the sun's illumination. Consider a binary image where pixels equaling 0 are black and 1 are white. Under these circumstances, the complement of the un-illuminated satellite image will be white. However, the noise level will also be complemented. If the sky in the field of view has a fraction of the total area (p_{stars}) containing stars, then the fraction of the background containing noise in the un-illuminated case is $1 - p_{stars}$. Since p_{stars} is typically $\ll 1$, this creates a high level of noise. To handle this, a layer within the boundary can be checked and penalized for the presence of noise. Simply check points parallel to the edge between vertices j and $j+1$, but inside the area. If a point is believed to be from the background, rather than the object, then perturb that point inward, and incorporate it in the list of line fit points. Let p_w be the width of a pixel, and n_{bl} be the desired boundary layer depth (in pixels). The steps are then as follows:

- For $i = 1$ to n_{bl}:
  - Create points $\vec{x} = \vec{d_j} + \alpha\vec{v} - ip_w\vec{u},\ \alpha \in [0, l_j]$.
  - If $p(I(\vec{x})|O) - p(I(\vec{x})|B) < 0$, then perturb the points as in (1), $\vec{x}+(p(I(\vec{x})|O)-p(I(\vec{x})|B))g\vec{u} \to \vec{x}$, and incorporate them in the data points used to fit a line between vertices j and $j+1$.

E. Step 5: Fit lines to these perturbed points along each edge.

This is a standard weighted least squares problem in the plane. The weighted least squares fit to the line $\vec{u}_j^T\vec{x} = b_j$ between vertices j and $j+1$ can be found from the points along this line, $\vec{x}_k$, by calculating

$$\bar{\vec{x}}_w = \frac{1}{n}\sum_{k=1}^{n} w_k\vec{x}_k$$

$$M = \sum_{k=1}^{n} w_k\vec{x}_k\vec{x}_k^T - n\bar{\vec{x}}_w\bar{\vec{x}}_w^T$$

where n is the number of perturbed points between vertices j and $j+1$ and w_k is the weight on the k^{th} perturbed point. The unit normal, $\vec{u}_j$, is then the eigenvector corresponding to the minimum eigenvalue of $\mathbf{M}$ and

$$b_j = \frac{n\bar{\vec{x}}_w^T\vec{u}_j}{\sum_{k=1}^{n} w_k} \tag{4}$$

Since M is only a 2×2 matrix, the eigenvalue problem can be explicitly solved to speed the calculations.

Points on the object are given a weight of one ($w_k = 1$), while any point statistical believed to be on the background ($p(I(\vec{x})|O) - p(I(\vec{x})|B) < 0$) is given much higher weight, w_b. Our experiments indicate that at least ten times higher weight on background points is useful for making the lines converge along the boundary.

F. Step 6: Create new vertices by intersecting non-collinear adjacent lines.

Let θ be the angle between normals $\vec{u}_j$ and $\vec{u}_{j+1}$. Then $\vec{u}_{j+1} = R_z\vec{u}_j$, where

$$R_z = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

is a rotation matrix about the "z" axis. Let

$$A_j = \begin{bmatrix} \vec{u}_j & \vec{u}_{j+1} \end{bmatrix} = \begin{bmatrix} \vec{u}_j & R_z\vec{u}_j \end{bmatrix}$$

The singular values of A_j are by definition the square root of the eigenvalues of $A_j^TA_j$. Because $\vec{u}_j$ is a unit vector, $A_j^TA_j$ can be written as

$$A_j^TA_j = I + \vec{u}_j^TR_z\vec{u}_jI^{\perp}$$

where

$$I^{\perp} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

The eigenvalues of $I^{\perp}$ are ± 1, thus the eigenvalues of $A_j^TA_j$ are $1 \pm \vec{u}_j^TR_z\vec{u}_j$. Since $\vec{u}_j$ is a unit vector and R_z is a rotation matrix, $\vec{u}_j^TR_z\vec{u}_j = \cos\theta$, thus the singular values of A_j are $\sqrt{1+\cos\theta}$, $\sqrt{1-\cos\theta}$. The condition number of A_j is the ratio of the maximum and minimum singular values,

$$cond(A_j) = \frac{\sqrt{1+\cos\theta}}{\sqrt{1-\cos\theta}} \tag{5}$$

For small angles between adjacent edges, A_j becomes very poorly conditioned ($\cos\theta \to 1 \implies cond(A_j) \to \infty$). As the next section will show, this implies from (6) that vertex $j+1$ will be poorly estimated. To avoid this problem and simultaneously combine edges that are nearly collinear, $\cos\theta$ can be found from $\cos\theta = \vec{u}_j^T\vec{u}_{j+1}$. If $\cos\theta < t$, where t is a threshold less than 1, then a new vertex is found by intersecting lines j and $j+1$. The new vertex, $\vec{d}_{j+1}$, is the intersection of the lines $\vec{u}_j^T\vec{x} = b_j$ and $\vec{u}_{j+1}^T\vec{x} = b_{j+1}$. Substituting $\vec{d}_{j+1} = \vec{x}$ into these two equations and solving for $\vec{d}_{j+1}$ yields

$$\vec{d}_{j+1} = \begin{bmatrix} \vec{u}_j & \vec{u}_{j+1} \end{bmatrix}^{-T}\begin{bmatrix} b_j \\ b_{j+1} \end{bmatrix} = A_j^{-T}\begin{bmatrix} b_j \\ b_{j+1} \end{bmatrix} \tag{6}$$

If the adjacent lines have nearly the same normal, but are translated, then instead of intersecting the two lines, place the new vertex at the point nearest the old vertex on the new line. That is, if $\cos\theta > t$ and $|b_j - b_{j+1}| > p_t$, then add the vertex $(I - u_{j+1}u_{j+1}^T)\vec{d}_{j+1} + b_{j+1}\vec{u}_{j+1}$. The constant p_t is a given threshold.

Because the new list of vertices enforce collinearity constraints, the distance between vertices may be much larger than l_d, and correspondingly, the number of vertices in the new list may be much smaller than the number of vertices found after step 3. This successive process of expanding, then contracting the number of vertices allows fine features to be extracted, while simultaneously automatically outputting the longest straight edges.

G. Step 7: Check for convergence, returning to step 2 until convergence takes place.

Convergence can be evaluated by finding the normed change in vertex locations. Once the number of vertices has stabilized, they can be stacked in a vector $\vec{d} = [\vec{d}_1^T\ \vec{d}_2^T\ \ldots \vec{d}_n^T]^T$. The weighted norm of the difference in this vector between iterations is a useful metric for determining convergence. Until it is sufficiently small, return to step 2.

H. Establishing the Correspondence

Once the straight line outline of the object is found using the above highly robust method, the correspondences between the vertices found in the image and those in a CAD model can be found. For satellites, a very coarse estimate of the pose is known from mission requirements. For instance, the solar arrays must face the sun. This knowledge can be used to find which vertices should appear in the image, and in what order.

III. Detection of Independently Moving but Visually Overlaid Objects.

There exists a need to detect an unwanted "chaser" spacecraft maneuvering or orbiting around a satellite. This problem can be broken into two cases:

1) The chaser's image is distinct from that of the satellite.
2) The chaser and satellite's images are blurred together, so they are visually overlaid.

In the first case, it is relatively straightforward to detect the presence of the unwanted chaser through a variety of motion analysis techniques. Consequently, it will not be considered herein. The second case, in contrast, can be difficult: are the changes in the image over time just due to motion of the satellite, or is a chaser present and accounting for some of the blurry image's changes? The image may be very blurry simply because of the distances involved, the available telescope, atmospheric effects, or poor (perhaps even totally missing) illumination. This section will develop a first level screening procedure which can very quickly indicate the presence of a possible chaser. Once this detection has occurred, further resources can be allocated (more processing, better telescopes, human visual inspection, better illumination and atmospheric seeing windows, etc.).

The new technique does not require detailed knowledge of the satellite's shape or motion, but is a simple method. More complex algorithms incorporating knowledge of satellite shape are derived in the next section. From successive images, this new, simple method finds the dimension of the subspace spanned by the motion in the image. If the dimension is higher than that explained by the satellite's motion alone, then a chaser may be present.

The new method is inspired by robotic motion segmentation algorithms [18]. Suppose the satellite has n points that are seen on all of the F image frames. Let the j^{th} satellite point be denoted as p_j ($j = 1 \ldots n$). Let g be the 4×4 homogeneous coordinate transformation from the satellite's coordinate system, S, to the camera's coordinate system, C. Then [5]

$$\begin{bmatrix} {}^C p_j \\ 1 \end{bmatrix} = g \begin{bmatrix} {}^S p_j \\ 1 \end{bmatrix} \tag{7}$$

where the prescripts C and S denote that the point is measured in the C and S coordinate systems, respectively. Denoting

$$\begin{bmatrix} p_j \\ 1 \end{bmatrix} = \bar{p}_j$$

equation (7) can be written more compactly as

$${}^C\bar{p}_j = g\,{}^S\bar{p}_j \tag{8}$$

Key fact: At a given time, *the same* g is used to transform *all* points on the satellite into camera coordinates.

Let the camera have a focal length of f, and let the satellite be at a distance z. If the camera's axis is the z axis, then the image plane position of p_j in the i^{th} frame is

$$d_{ij} = \frac{f}{z} \begin{bmatrix} I_2 & 0 \end{bmatrix} {}^C\bar{p}_j$$

Note that this equation assumes that the distance from the camera to the satellite is much larger than the size of the satellite, so minor differences in the distance from the camera to a point on a satellite caused by that point's location on the satellite are irrelevant (affine camera model). The affine camera model is highly accurate when imaging objects from Earth to space, as the distance is hundreds of kilometers even for low Earth orbits. From (8),

$$d_{ij} = \frac{f}{z} \begin{bmatrix} I_2 & 0 \end{bmatrix} g_i\,{}^S\bar{p}_j \tag{9}$$

where the i subscript on g denotes the homogeneous coordinate transformation when the i^{th} frame is taken. Denote the 2×4 transformation matrix as

$$M_i = \frac{f}{z} \begin{bmatrix} I_2 & 0 \end{bmatrix} g_i$$

Equation (9) then becomes

$$d_{ij} = M_i\,{}^S\bar{p}_j$$

Let the n points be tracked from one frame to the next over $i = 1 \ldots F$ frames. Then the i^{th} row of the data matrix, $D = \{d_{ij}\}$ is

$$M_i \begin{bmatrix} {}^S\bar{p}_1 & {}^S\bar{p}_2 & \cdots {}^S\bar{p}_n \end{bmatrix}$$

Denote the $4 \times n$ post-multiplying matrix as ${}^S P$,

$$ {}^S P = \left[{}^S \bar{p}_1 \ {}^S \bar{p}_2 \ \cdots {}^S \bar{p}_n \ \right] $$

Finally,

$$ D = \underbrace{\begin{bmatrix} M_1 \\ M_2 \\ \vdots \\ M_F \end{bmatrix}}_{2F \times 4} \underbrace{{}^S P}_{4 \times n} \tag{10} $$

Equation (10) shows that the data matrix, $D = \{d_{ij}\}$ can be decomposed into two matrices multiplied together. Since image data is two dimensional, the first matrix has dimension $2F \times 4$, while ${}^S P$ is $4 \times n$. As the rank of a matrix is less than or equal to the smallest dimension of its factors, the rank of D is less than or equal to four. This is a surprising result, as D may be a very large matrix if many points are tracked across many frames. If only rotations are present, then a similar analysis can be used to show that the rank of D is less than or equal to three.

When the real data is collected, some of the points may not in actuality be located on the satellite–they could, for instance, lie on a chaser vehicle. In this case, the above arguments no longer hold, and the dimension of D can exceed four. This implies that the dimension of the data matrix can be used as a detection criteria–if it exceeds four, then a chaser may possibly be present, so further investigations are merited.

Note that this detection algorithm is very simple to implement, as it requires no knowledge of the satellite. Moreover, point correspondences are only required between frames. This is comparatively simple, since image points typically don't move much over a single sample time, thus finding two corresponding points is simplified. To implement the algorithm, the measurements must simply be collected in a single matrix, D, where each column contains the image positions of a single satellite point as it appears in different images. The rank of D is calculated to detect multiple moving bodies.

IV. Detection of Extraneous or Missing Objects

The simple technique in Section III is very useful for many scenarios, but it can be improved by including more knowledge. This section assumes that a solid model of the satellite's shape is available, and uses that knowledge to compare the obtained image with predictions of what the image should look like. Gross differences indicate an anomaly.

This section briefly reviews pose estimation results found in [12]. The proofs and further details (including methods for evaluating the estimation error and viewpoint) are in [12]. From a noisy, diffraction limited image of a known satellite in an unknown pose, the pose will be estimated. The CAD model can then be placed in the same pose and projected onto two dimensions to create a synthetic prediction of what the satellite's image should look like. Subtraction of the actual image from the synthetic image produces a signal useful for automatically checking any satellite anomalies.

First, the estimation of orientation alone will be treated. Define the hat function $\hat{(\cdot)}$ as the cross product matrix, i.e.

$$ \hat{\vec{z}} = \begin{bmatrix} 0 & -z_3 & z_2 \\ z_3 & 0 & -z_1 \\ -z_2 & z_1 & 0 \end{bmatrix} $$

Theorem *1:* Given $\vec{z}_i$, $\vec{v}_i \in \mathbb{R}^3$ and $w_i \in \mathbb{R}_+$, $i = 1 \ldots n$, the minimum of

$$ J = \sum_{i=1}^{n} w_i ||R\vec{z}_i - \alpha \vec{v}_i||^2 \tag{11} $$

over $R \in$ SO(3), $\alpha \in \mathbb{R}_+$ is found by first calculating

$$ D = \sum_{i=1}^{n} w_i K(\vec{v}_i, \vec{z}_i) \tag{12} $$

where K is the bilinear, symmetric matrix function

$$ K(\vec{v}, \vec{z}) = \begin{bmatrix} \vec{v}\vec{z}^T + \vec{z}\vec{v}^T - 2\vec{v}^T\vec{z} I & \hat{\vec{z}}\vec{v} \\ (\hat{\vec{z}}\vec{v})^T & 0 \end{bmatrix} \tag{13} $$

The eigenvector, $\vec{\epsilon}$, of D corresponding to the maximum eigenvalue is the unit quaternion representation of the optimal R. The optimal scaling is

$$ \alpha = \frac{\sum_{i=1}^{n} w_i \vec{v}_i^T R \vec{z}_i}{\sum_{i=1}^{n} w_i ||\vec{v}_i||^2} \tag{14} $$

□

Orientation estimation from 2-d data does not enjoy the invariance to scaling found in Theorem 1, so the case where scaling is known will be solved, then scaling will be estimated separately.

Proposition *2:* Given $\vec{z}_i \in \mathbb{R}^3$, $\vec{d}_i \in \mathbb{R}^2$, and $w_i \in \mathbb{R}_+$, $i = 1 \ldots n$, $P = [I_2 \ 0]$ the minimum of

$$ J_{2d} = \sum_{i=1}^{n} w_i ||PR\vec{z}_i - \vec{d}_i||^2 \tag{15} $$

over $R \in$ SO(3), is found by first calculating

$$ \vec{v}_i = \begin{bmatrix} \vec{d}_i \\ \cdots \\ \pm\sqrt{||\vec{z}_i||^2 - ||\vec{d}_i||^2} \end{bmatrix}, \ i = 1 \ldots n \tag{16} $$

Next, the 2^n 3-d orientation problems corresponding to all possible v_i, $i = 1 \ldots n$ $\pm$ sign choices are solved using Theorem 1:

$$ \min_{R \in SO(3)} \sum_{i=1}^{n} w_i ||R\vec{z}_i - \vec{v}_i||^2 $$

From these 2^n solutions, the optimum is that minimizing J_{2d}.

□

Lemma 3 will now show how these results can be brought together to estimate orientation from 2-d measurements with unknown scaling.

Lemma *3:* Given $\vec{z}_i \in \mathbb{R}^3$, $\vec{d}_i \in \mathbb{R}^2$, and $w_i \in \mathbb{R}_+$, $i = 1 \ldots n$, $P = [I_2 \ 0]$ the minimizer of

$$ J = \sum_{i=1}^{n} w_i ||PR\vec{z}_i - \alpha \vec{d}_i||^2 \tag{17} $$

over $R \in$ SO(3), $\alpha \in \mathbb{R}_+$, is approximated by first estimating α by $\alpha_e = \frac{1}{\max_{i=1\ldots n} ||\vec{d}_i||/||\vec{z}_i||}$. R can then be estimated by replacing all measurement vectors $\vec{d}_i$ with scaled versions $\vec{d}_i \mapsto \alpha_e \vec{d}_i$, $i = 1 \ldots n$, and applying Prop. 2.

□

Finally, to estimate both orientation and position, use Prop. 4.

Proposition *4:* Given data points $\vec{z}_i \in \mathbb{R}^3$ and corresponding sensed points $\vec{q}_i \in \mathbb{R}^2$, along with weightings $w_i \in \mathbb{R}_+$, $i = 1 \ldots n$, $P = [I_2\ 0]$, the minimizer of

$$J = \sum_{i=1}^{n} w_i ||P[R\vec{z}_i + \vec{p}] - \alpha\vec{q}_i||^2 \tag{18}$$

over $g = (R, \vec{p}) \in$ SE(3) ($R \in$ SO(3), $\vec{p} \in \mathbb{R}^3$), and $\alpha \in \mathbb{R}_+$ can be estimated by forming the free vectors $\vec{z}_{ij} = \vec{z}_i - \vec{z}_j$, $\vec{d}_{ij} = \vec{q}_i - \vec{q}_j$, $i, j = 1 \ldots n$. R and α can then be estimated using Lemma 3 by replacing $\vec{z}_i$ by $\vec{z}_{ij}$ and $\vec{d}_i$ by $\vec{d}_{ij}$. The first two components of $\vec{p}$ are estimated by

$$\begin{bmatrix} p_x \\ p_y \end{bmatrix} = \vec{p}_a = P\vec{p} = \frac{\alpha \sum_{i=1}^{n} w_i \vec{q}_i - R_a \sum_{i=1}^{n} w_i \vec{z}_i}{\sum_{i=1}^{n} w_i} \tag{19}$$

where R_a denotes the 2×3 matrix consisting of the first two rows of R. When the measurements arise from an optical system with focal length f, and p_z denotes the third component of $\vec{p}$, and if the imaging is affine ($||\vec{z}_i|| \ll p_z$ for all points $i = 1 \ldots n$), then

$$p_z \approx \alpha f \tag{20}$$

□

V. Simulation Results

Figures 1 and 2 illustrate the silhouettes obtained using a standard gradient (Sobel) operator for both the sun illuminated and un-illuminated (only back lit) cases. While the Sobel operator performs well when the satellite is illuminated, it completely fails without illumination.

The new SSLS method, on the other hand, is able to find the silhouette (Fig. 3). Table 4 documents the parameters used in this simulation.

Figure 5 illustrates a sequence of diffracted images containing both the satellite and a chaser in a nearby relative orbit. The + signs indicate points that are tracked between each image, while the circled + signs indicate those points due to the chase vehicle. The satellite's translational motion is tracked, so only rotations are present in the image. A data matrix, D, is formed where each column lists the image plane positions of a single point. Since only rotations are present, a single rigid body ideally has rank(D) of at most three. When all eight columns are included, the singular values are: $(347.28, 75.33, 31.48, 3.35, 0.69, 0.51, 0.23, 0.14)$. If the two columns corresponding to points on the chaser are excluded, then the singular values become: $(307.55, 73.49, 1.32, 0.63, 0.48, 0.19)$. In the later case, the rank should be at most three, since all the data points stem from a single rigid body. However, the rank is greater than three in this realistic simulation due to noise and imperfect correspondences. Nevertheless, the third and higher singular values in the multiple body case are significantly larger than those in the later, single body case–a discernable difference that can be wisely used as an indication of multiple body presence.

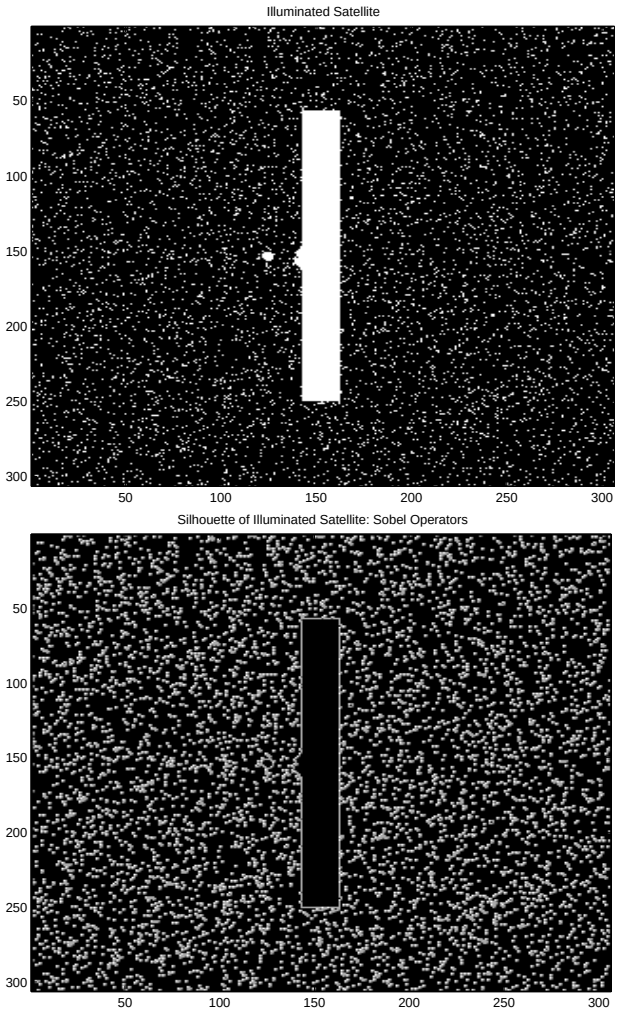

Fig. 1. The illuminated satellite (top), and its silhouette found using gradient masks (bottom).

From a noisy, diffraction limited image of a known satellite in an unknown pose, the pose will be estimated. The CAD model can then be placed in the same pose and projected onto two dimensions to create a synthetic prediction of what the satellite's image should look like. Subtraction of the actual image from the synthetic image produces a signal useful for automatically checking any satellite anomalies.

Fig. 6 depicts a hypothetical satellite. The far corners of the solar arrays are chosen as the data points ($\vec{z}_i$, $i = 1, \ldots, 4$, depicted as 0's), as they produce an easily identifiable signature. As the satellite moves, several images are captured (Fig. 7). The pose will be estimated and the satellite inspected for images 1 and 4 (1 is the leftmost).

Because satellites orbit at significant distances from the Earth, the actual image is degraded appreciably by diffraction effects (see Fig. 8). Using a binary thresholding technique, it

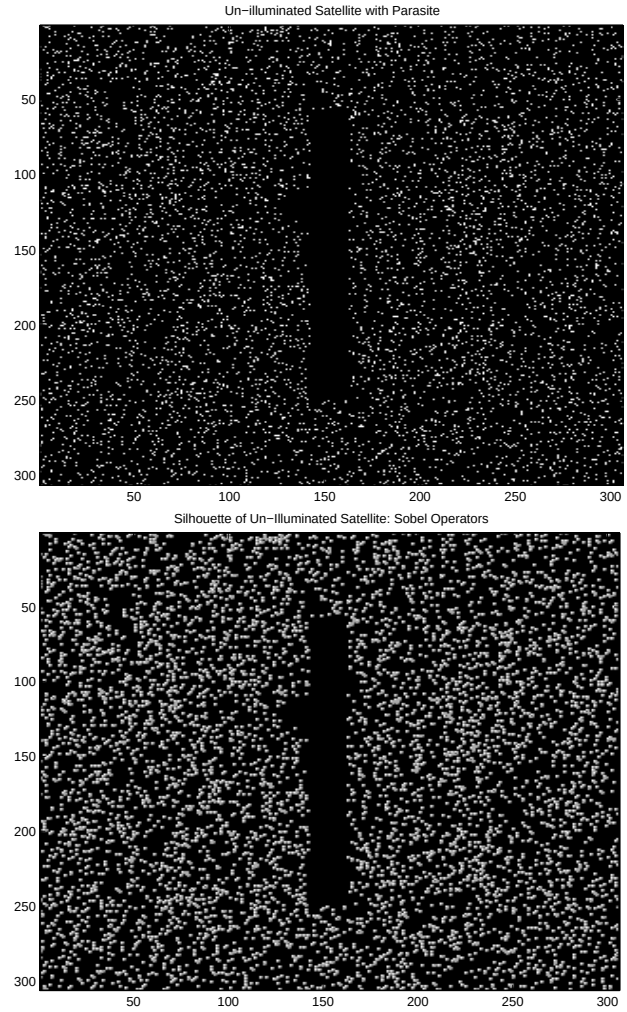

Fig. 2. The un-illuminated satellite image (top), and its gradient (bottom). The silhouette is not found.

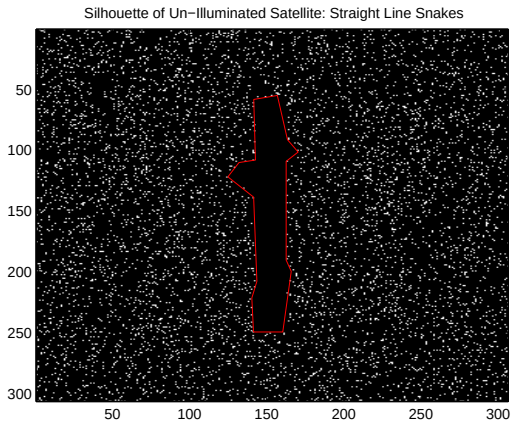

Fig. 3. The silhouette is found using the SSLS method.

n_{its}	l_d	l_{min}	n_{bl}	w_b	t	g	p_t
1400	20	5	3	100	$\cos(7^o)$	0.5	∞

Fig. 4. Parameters used in the SSLS method.

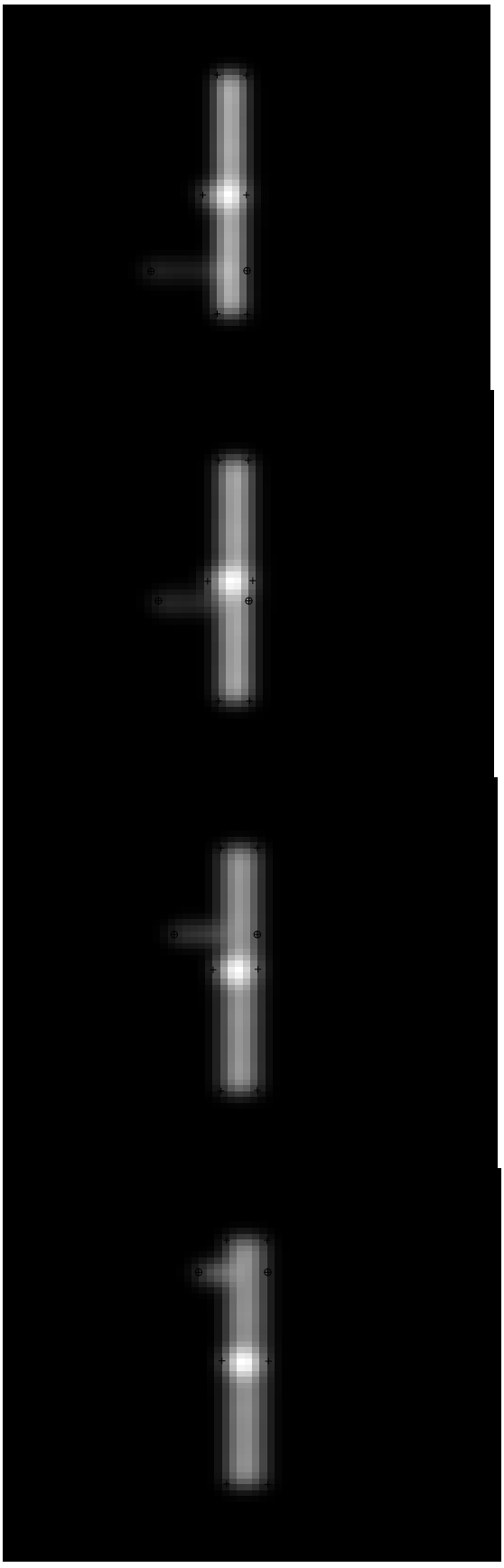

Fig. 5. A sequence of diffracted images, and the tracked correspondence points.

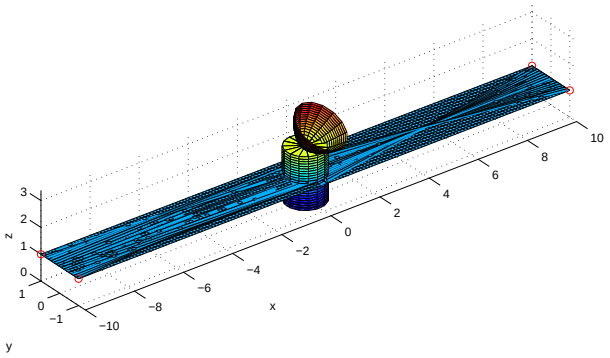

Fig. 6. CAD model of the imaged satellite, with O's denoting the point correspondences.

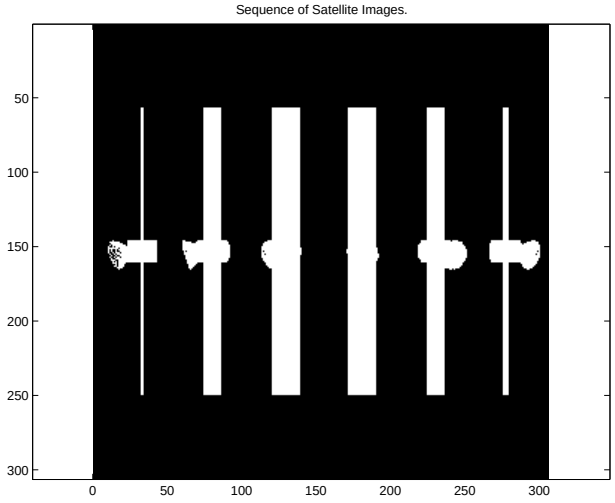

Fig. 7. As the satellite sweeps across the sky, several images are captured.

is enhanced, and the solar array corners in the image ($\vec{q}_i$) are found (Fig. 9).

Despite the effects of diffraction, thresholding, and pixel round-off, the pose estimate is quite accurate, producing an angular estimation error of only $e_\theta = 5.3^o$. A thresholded version of the predicted image is subtracted from the thresholded measured image (Fig. 9) to produce the inspection image (Fig. 10). Although it suffices for our present purposes, an improved match can certainly be obtained from this subtracted image by employing, for example, the methods in [7] to fine tune the estimate.

Automatically determining the point correspondences $\vec{q}_i$ can be difficult. These simulations first find a point inside the satellite by a very coarse template matching strategy. Then, a statistical snake [14] grown from that initial point robustly identifies the perimeter. Finally, vertices along that perimeter are extracted (see [13] for further details). Figures 11 and 12 illustrate the results on image 4. In this case, $e_\theta = 25^o$.

VI. Conclusions

This paper develops three new, interconnected techniques useful for the autonomous distant visual surveillance of satellites. First, silhouetting of man made, erratically illuminated

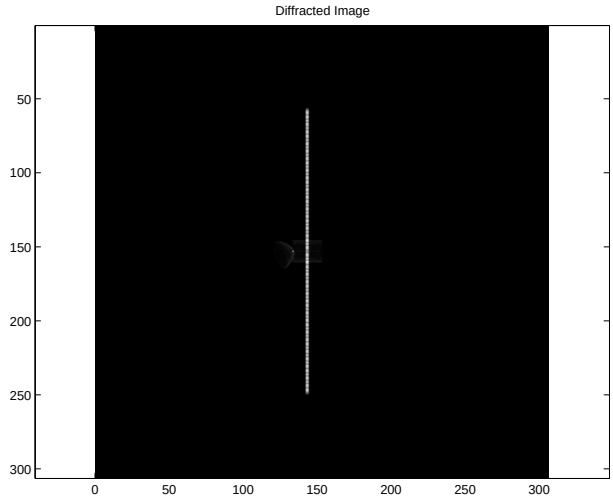

Fig. 8. The first raw, diffracted image.

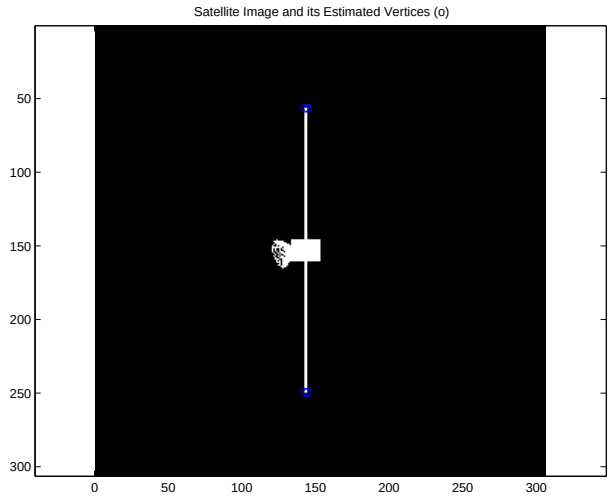

Fig. 9. The first image with the sensed points, $\vec{q}_i$, denoted by (0).

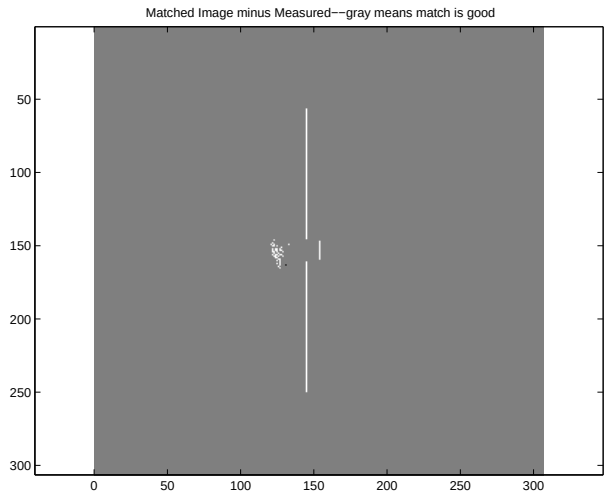

Fig. 10. Combining pose estimates with the CAD model, a predicted image is calculated and subtracted from the measured image.

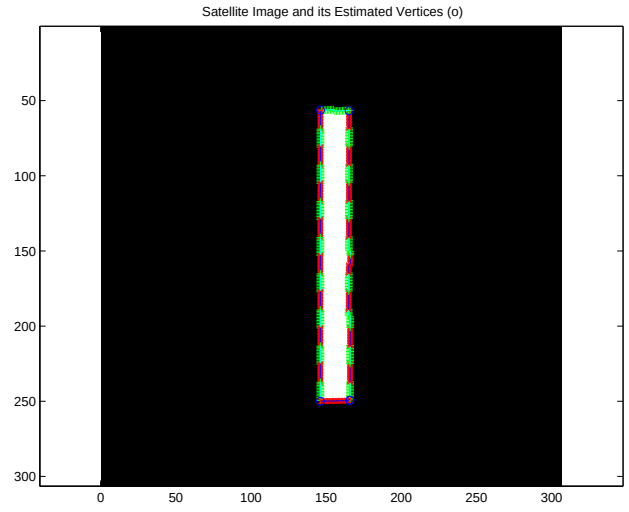

Fig. 11. The fourth image, with its points $\vec{q}_i$ denoted by (0). The outside perimeter is identified by statistical snakes.

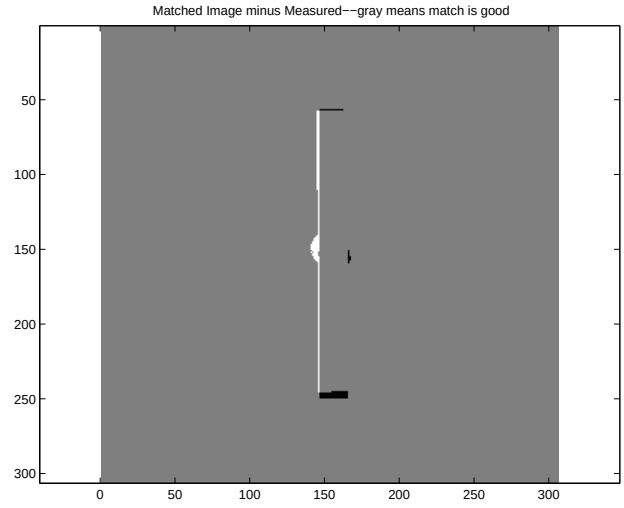

Fig. 12. Combining pose estimates with the CAD model, a predicted image is calculated and subtracted from the fourth measured image.

satellites is performed. Illumination cases include full sun from an arbitrary (often awkward) viewing angle and un-illuminated (back-lit by the star field). New Statistical Straight Line Snakes (SSLS) prove efficient in finding the silhouette, even in the un-illuminated case. In contrast, gradient techniques based on the Sobel operator fail utterly. The silhouette is composed of straight line segments, which are easy to calculate, fit the straight lines inherent in man made objects, and lend themselves to further processing (pose estimation, template matching, etc.) A current silhouette can be used as an initial guess to speed silhouette tracking between successive images. Once the silhouette has been used to find correspondence points, a second method for detecting a moving, nearby chaser vehicle is derived. The hard case is treated in which the chaser and satellite are so nearby that their images are blurred together. The algorithm finds the dimension of motion generated by the sequence of images. If the dimension is higher than that explained by a single rigid body, then this indicates a possible chaser. No knowledge of the satellite or chaser's shape or motion is required. Simulations indicate discernable differences in the singular values obtained with one versus two bodies present.

Independent relative motion between the satellite and chaser is required–if the chaser is immobile with respect to the satellite, then a third technique must be used. This third method incorporates the satellite's solid model to estimate its pose from a noisy, diffraction limited image. The pose is then combined with the solid and optical model to create synthetic expected images. Inspection is performed by comparing these with the actual images. The new pose algorithm first estimates depth by a least upper bound technique. A fast method is derived of optimally estimating the rotation matrix by a sequence of analytical solutions (rather than a nonlinear numerical optimization!). Simulations illustrate a satellite inspection from Earth to low Earth orbit.

REFERENCES

[1] Wael Abd-Almageed, C. Smith, and Samah Ramadan. Kernel snakes: Non-parametric active contour models. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, 2003.

[2] Adnan Ansar and Kostas Daniilidis. Linear pose estimation from points or lines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):578–589, May 2003.

[3] A. Chiuso, P. Favaro, H. Jin, and S. Soatto. Motion and structure casually integrated over time. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):523–535, May 2002.

[4] A.I. Comport, D. Kragic, E. Marchand, and F. Chaumette. Robust real-time visual tracking: Comparison, theoretical analysis and performance evaluation. In *IEEE International Conference on Robotics and Automation*, pages 2852–2857, Barcelona, Spain, April 2006.

[5] John J. Craig. *Introduction to Robotics: Mechanics and Control*. Pearson Prentice Hall, 3 edition, 2005.

[6] D. DeMenthon and L.S. Davis. Exact and approximate solutions of the perspective three-point problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(11):1100–1105, November 1992.

[7] T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):932–946, July 2002.

[8] P.D. Fiore. Efficient linear solution of exterior orientation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:140–148, 2001.

[9] K.S. Fu, R.C. Gonzalez, and C.S.G. Lee. *Robotics: Control, Sensing, Vision, and Intelligence*. McGraw–Hill, New York, New York, 1987.

[10] Robert M. Haralick and Linda G. Shapiro. *Computer and Robot Vision*, volume 1. Addison-Wesley, Reading, MA, 1992.

[11] C. Lu, G. Hager, and E. Mjolsness. Fast and globally convergent pose estimation from video images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(6):610–622, June 2000.

[12] J.E. McInroy, R.S. Erwin, and L.M. Robertson. Rapid, robust, optimal pose estimation from a single affine image. *In Preparation for the IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(x):x–x, May 2007.

[13] J.E. McInroy, L.M. Robertson, and R.S. Erwin. Distant visual detection of satellite anomalies. In *Preparation for the IEEE International Conference on Robotics and Automation*, April 2007.

[14] D. Perrin and C. Smith. Rethinking classical internal forces for active contour models. In *IEEE International Conference on Computer Vision and Pattern Recognition*.

[15] L. Quan and Z. Lan. Linear n-point camera pose determination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21:774–780, 1999.

[16] H. Rehbinder and B.K. Ghosh. Pose estimation using line-based dynamic vision and inertial sensors. *IEEE Transactions on Automatic Control*, 48(2):186–199, February 2003.

[17] J.P. Tarel and D. B. Cooper. The complex representation of algebraic curves and its simple exploitation for pose estimation and invariant recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(7):663–674, July 2000.

[18] R. Vidal. Multi-subspace methods for motion segmentation from affine, perspective, and central panoramic cameras. In *IEEE International Conference on Robotics and Automation*, pages 1228–1233, Barcelona, Spain, April 2006.